

УДК 621.391:681.5

ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ РОЗПОДІЛЕНИХ ІНФОКОМУНІКАЦІЙНИХ СИСТЕМ НА ОСНОВІ ОБРОБЛЕННЯ ВЕЛИКИХ ОБСЯГІВ ДАНИХ

DOI 10.36994/2707-4110-2019-2-23-03

Климаш М.М., д.т.н., проф., Національний університет «Львівська політехніка», Львів, Україна. mklimash@polynet.lviv.ua

Гордійчук-Бублівська О.В., Національний університет «Львівська політехніка», Львів, Україна. obublivska@gmail.com

Чайковський І.Б., к.т.н., Національний університет «Львівська політехніка», Львів, Україна. ihor.b.chaikovskyi@lpnu.ua

Урікова О.М., к.е.н., Національний університет «Львівська політехніка», Львів, Україна. oksana.m.urikova@lpnu.ua

Анотація. В роботі досліджено особливості обробки великих масивів інформації для розподілених систем. Застосовано метод сингулярної декомпозиції даних, завдяки якому можна зменшити обсяг оброблюваних даних, відкинувши надлишковість. Отримано залежності ефективності обчислень у розподілених системах із використанням протоколу обміну повідомленнями MPI та програмної моделі взаємодії вузлів MapReduce. Проаналізовано ефективність застосування кожної технології для обробки масивів даних різних розмірів. Визначено, що протокол MPI дозволяє ефективніше проводити обчислення невеликих обсягів інформації. При збільшенні масивів даних доцільно застосовувати модель Map Reduce.

В системах з розподіленими ресурсами пристрій обробляє лише певну частину даних, які надходять для обчислення. Після того, як окремі обчислювальні засоби проведуть опрацювання своїх задач, відбувається об'єднання всіх частин та отримується кінцевий результат. Далі пристрої отримують частини інших даних і т.д. Внаслідок такого підходу підвищується продуктивність системи, оскільки значно швидше опрацьовуються великі обсяги інформації. Розподілені обчислювальні системи мають ряд інших переваг, зокрема, масштабованість і надійність обробки та зберігання даних. В подібних системах дані розподіляються різними пристроями невеликими частинами, що значно зменшує втрату інформації при помилках і пошкодженні останньої.

Звичайні нерозподілені системи обробки даних неефективні для великого обсягів інформації через невисоку продуктивність обчислень. Пропонується використовувати розподілені системи, в яких використовують метод сингулярної декомпозиції даних, що дозволить зменшити обсяг оброблюваної інформації.

В результаті дослідження систем з використанням протоколу MPI та моделі MapReduce отримано залежності тривалості обчислень від кількості процесів, які свідчать про доцільність використання розподілених обчислень при обробці великих масивів даних. Також визначено, що розподілені системи із застосуванням моделі обчислень MapReduce працюють значно ефективніше ніж за протоколом MPI, особливо при великому обсягу даних.

Ключові слова: Розподілені системи, Big Data, Сингулярна декомпозиція даних, MPI, MapReduce.

INVESTIGATION OF THE EFFICIENCY OF DISTRIBUTED INFORMATION SYSTEMS BASED ON THE PROCESSING OF LARGE AMOUNTS OF DATA

Mykhajlo Klymash, Dr.habil., Prof., Lviv Polytechnic National University, Lviv, Ukraine. mklimash@polynet.lviv.ua

Olena Hordiichuk — Bublivska, Lviv Polytechnic National University, Lviv, Ukraine. obublivska@gmail.com

Ihor Tchaikovskiy, Ph.D., Lviv Polytechnic National University, Lviv, Ukraine. ihor.b.chaikovskyi@lpnu.ua

Oksana Urikova, Ph.D., Lviv Polytechnic National University, Lviv, Ukraine. oksana.m.urikova@lpnu.ua

Abstract. The features of processing large arrays of information for distributed systems are investigated. The method of singular data decomposition is applied, which can be used to reduce the amount of data processed by discarding excess data. Dependencies of computational efficiency on distributed systems were obtained using the MPI messaging protocol and the MapReduce nodal interaction model. The efficiency of the application of each technology for the processing of data sets of different sizes is analyzed. It has been determined that the MPI protocol enables more efficient computation of small amounts of information. As the data arrays increase, it is advisable to use the Map Reduce model.

On shared resource systems, the device processes only some of the data that is being received for calculation. After the individual calculators complete their tasks, all parts are combined and the result is obtained. Further devices receive portions of other data, etc. This approach improves system performance because large amounts of information are processed much faster. Distributed computing systems have a number of other advantages, in particular, the scalability and reliability of data processing and storage. In such systems, the data is distributed by different devices to small parts, which significantly reduces the loss of information in the event of errors and damage to the latter

Conventional retained data processing systems are inefficient for large amounts of information due to poor computing performance. It is proposed to use distributed systems that use the method of singular decomposition of data, which will reduce the amount of information processed.

The study of systems using the MPI protocol and the MapReduce model obtained the dependence of the duration of the calculations on the number of processes, which testify to the expediency of using distributed computing in the processing of large data

sets. It has also been found that distributed systems using the MapReduce model work much more efficiently than MPI, especially with large amounts of data.

Keywords: Distributed System, Big Data, Singular Value Decomposition, MPI, MapReduce.

Вступ

На даний час часто необхідно залучати потужні засоби для обробки інформації. Це відбувається через те, що проводиться все більше досліджень, які вимагають складних та високоточних обчислень.

Основною характеристикою роботи пристроїв обробки даних є продуктивність, тобто кількість арифметичних операцій, які виконуються за одиницю часу. Також за продуктивністю оцінюють ефективність роботи обчислювальних систем.

В системах з розподіленими ресурсами пристрій обробляє лише певну частину даних, які надходять для обчислення. Після того, як окремі обчислювальні засоби проведуть опрацювання своїх задач, відбувається об'єднання всіх частин та отримується кінцевий результат. Далі пристрої отримують частини інших даних і т.д. Внаслідок такого підходу підвищується продуктивність системи, оскільки значно швидше опрацьовуються великі обсяги інформації. Розподілені обчислювальні системи мають ряд інших переваг, зокрема, масштабованість і надійність обробки та зберігання даних. В подібних системах дані розподіляються різними пристроями невеликими частинами, що значно зменшує втрату інформації при помилках і пошкодженні останньої [1–3].

Оскільки, кількість даних, які потребують обробки, невинно зростає і для їх обчислення недостатньо існуючих засобів, навіть якщо вони мають дуже високу продуктивність, запропоновано застосовувати обчислювальні системи з розподілом ресурсів.

Алгоритми обробки даних в розподілених інфокомунікаційних системах

Рекомендаційні системи. Для опрацювання великих даних різноманітного типу часто застосовують рекомендаційні системи обробки, які містять дані про вподобання користувачів, отримані з різних джерел, про їхні дії, пошукові запити та огляд рекламних оголошень та формують матрицю вподобань (Рис. 1).

На основі матриць вподобань визначають товари, що найбільше зацікавили користувача: рекомендуються товари, найбільш схожі до тих, що раніше отримали високу оцінку або шукають схожих користувачів та рекомендують вподобані товари один одному.



	Товар 1	Товар 2	Товар 3	Товар 4
Користувач 1	5	4	?	1
Користувач 2	2	1	4	3
Користувач 3	3	3	3	?
Користувач 4	5	4	3	4

Рис. 1. Матриця вподобань користувачів

Найчастіше в рекомендаційних системах для обробки даних використовують метод сингулярної декомпозиції — SVD (Singular Value Decomposition). Завдяки ньому можна понизити ранг матриць даних.

Сингулярний розклад даних. Алгоритм SVD розкладає вхідну матрицю M на добуток трьох матриць: U ($m \times m$), Σ ($m \times n$) та V ($n \times n$). Матриці U та V ортогональні, (тобто $U^*U = I_m$, $V^*V = I_n$, U^* і V^* -транспоновані матриці U і V ; I_m , I_n — одиничні матриці). Σ — діагональна матриця (всі елементи, які не розташовані на головній діагоналі, рівні нулю) (Рис. 2).

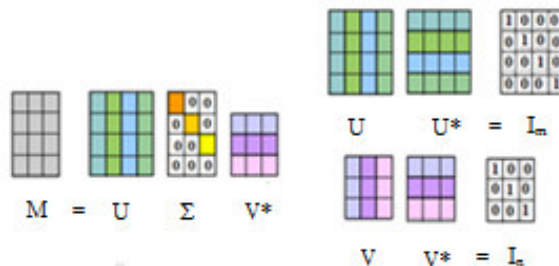


Рис. 2. Сингулярний розклад матриці даних (за Wikipedia)

В матриці Σ по діагоналі містяться елементи в порядку зростання. Для зниження рангу матриці Σ можна обрати k найбільших діагональних значень, а решту — відкинути. В матрицях U та V теж залишаються перші k рядків (Рис. 3).

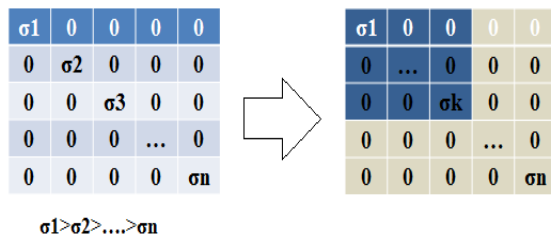


Рис. 3. Обробка матриці Σ

Після перемноження матриць U_1 , Σ_1 та V_1 отримуємо матрицю M_1 рангу k , яка з високою точністю повторює характеристики матриці M (Рис. 4).

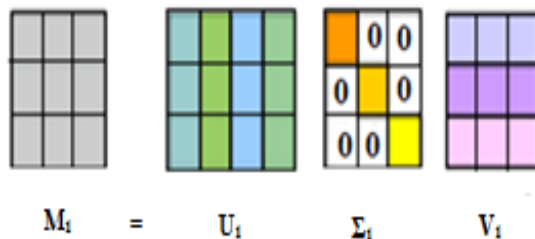


Рис. 4. Зменшення матриці даних в алгоритмі SVD

Завдяки цьому зменшується обсяг даних, які слід обробити.

Протокол MPI. Message Passing Interface — протокол обміну повідомленнями між вузлами. Застосовується в розподілених системах для комунікації між обчислювальними процесами. Для проведення розподілених обчислень один вузол в мережі проводить розподіл даних на частини, відповідно до кількості інших вузлів. Дані надсилаються на вузли за допомогою повідомлення Send, обробляються та відсилаються на керуючий вузол, де збираються до купи (Рис. 5).



Рис. 5. Архітектура протоколу MPI

Перевагою використання протоколу в розподілених системах є висока надійність, оскільки доки кожен обчислювальний процес не відправить опрацьовані дані на головний вузол, робота алгоритму не буде завершена [4].

Програмна модель MapReduce. MapReduce застосовується для паралельної або розподіленої обробки інформації. Впроваджена корпорацією Google для того, щоб ефективно опрацьовувати значні обсяги даних, використовуючи кластери комп'ютерів. Принцип роботи моделі наведено на рис. 6. В MapReduce дані представлені як спеціальні записи. Опрацювання інформації здійснюється в три етапи. Спочатку за допомогою визначеної користувачем функції Map вхідні дані фільтруються та попередньо опрацьовуються. Потім відбувається етап Shuffle, непомітний користувачам, на якому дані, отримані зі стадії Map, розділяються на менші частини та отримують ключ. На етапі Reduce виконується об'єднання результатів обробки даних, при цьому групуючи дані з однаковим ключем. Також, на етапі Map здійснюється формування пар «ключ — значення» для кожного вхідного запиту (Рис. 6).

Для практичної реалізації моделі MapReduce використовують платформу Apache Hadoop, яка призначена для розподілення та зберігання даних великих обсягів. Hadoop проводить поділ даних на менші частини, що потім можуть обчислюватися на окремих вузлах певного комп'ютерного кластера.

Ядро технології складається із розподіленої файлової системи (HDFS — Hadoop Distributed File System) та обчислювальної системи (що базується на MapReduce) [5–7].

На відміну від MPI, повідомлення про відправку і отримання даних не надсилаються, що значно спрощує роботу системи. Дані розсилаються просто усім доступним обчислювальним пристроям і записуються в їх пам'ять, а після опрацювання видаляються. Через це процес обчислень сповільнюється.

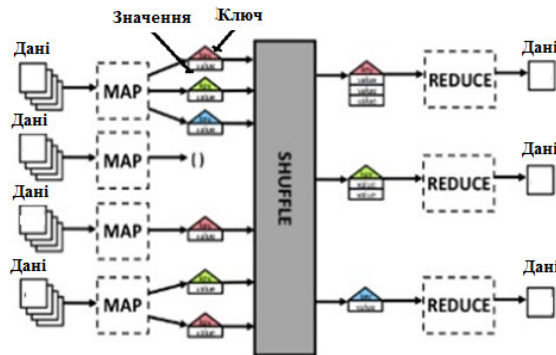


Рис. 6. Програмна модель MapReduce

Дослідження ефективності обробки даних в розподілених системах

Для проведення обчислень використано мову програмування Java та середовище розробки IntelliJ IDEA.

Обчислення сингулярної декомпозиції даних у системах із використанням MPI. Було реалізовано стандартний алгоритм пошуку сингулярного розкладу матриці, описаний в попередньому розділі. В ході експерименту досліджено залежності часу виконання програми від способу організації обчислень та розмірності даних. При розрахунках застосовано бібліотеки OpenMpi та Jama.

На Рис.7 представлено результат експерименту. Приведена залежність часу виконання сингулярного розкладу матриці розмірністю 120 рядків та 120 стовпців в залежності від кількості обчислювальних процесів. Можна зробити висновок, що при збільшенні кількості процесів зменшується час виконання, обчислень оскільки дані розділяються на менші за розміром частини, що швидше обробляються процесором.

На Рис. 8 представлена залежність часу виконання обчислень в залежності від розміру масиву обчислюваних даних при обробці в двохпроцесорній системі.

Зі збільшенням розмірності даних час обчислень зростає. Для того щоб більш ефективно проводити обчислення, слід збільшити кількість процесорів, як це було показано на Рис. 7. З результатів проведених експериментів видно, що час виконання програмного алгоритму зростає зі зменшення кількості обчислювальних пристроїв. Для однопроцесорної системи, яка, отже, не є розподіленою, час обчислень найбільший.

Порівняння ефективності застосування MPI для розподіленого обчислення різних обсягів даних. Проведемо експерименти по обчисленню SVD розкладу матриці для матриць різних розмірів, використовуючи різну кількість процесних пристроїв в розподіленій системі.

1) Згенеруємо матрицю розміру 20 рядків на 20 стовпців. Такий обсяг даних можна вважати малим. Кожен елемент матриці має тип double (8 байтів).

Тоді загальний обсяг становитиме $8 \times 20 \times 20 = 3200$ байт = 3,2 кбайтів. Кількість процесорів послідовно змінюється: від 1 до 32. Результати експерименту на Рис. 9.

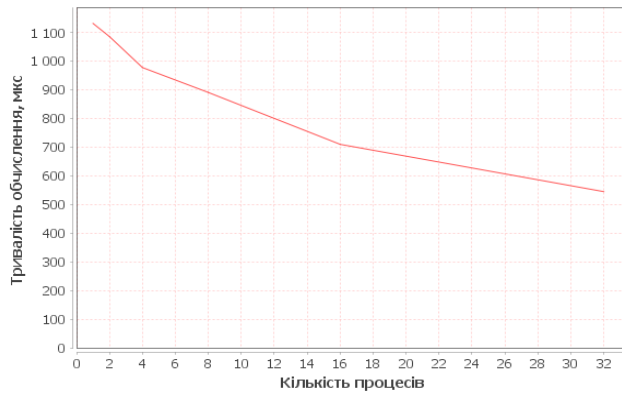


Рис. 7. Залежність часу виконання обчислень в залежності від кількості обчислювальних процесів

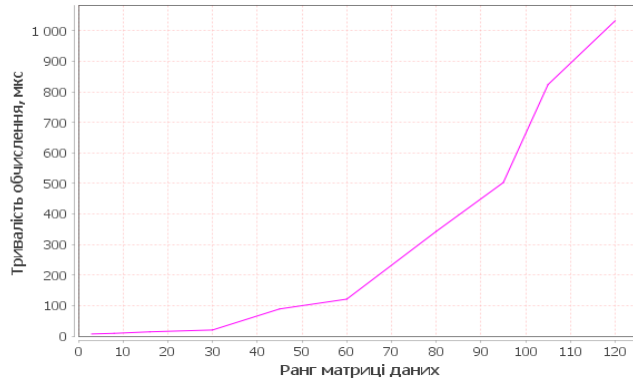


Рис. 8. Залежність часу виконання розподілених обчислень від розмірності даних в системі з двома обчислювальними пристроями

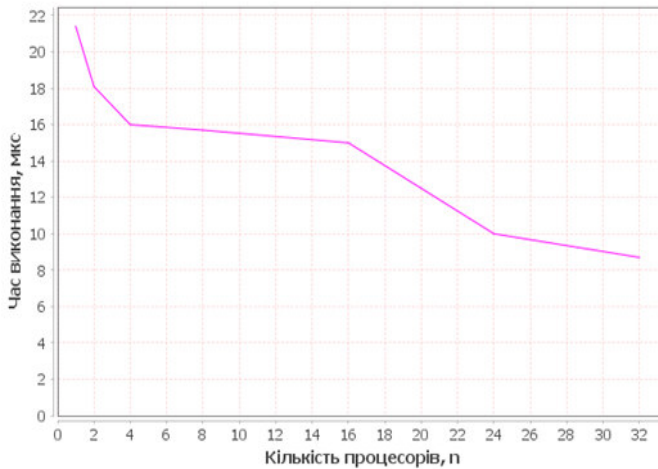


Рис. 9. Залежність часу виконання обчислень в залежності кількості обчислювальних процесів при розмірі матриці 20 x 20

З графіку можна зробити висновок, що зі збільшення кількості процесорів час обчислень зменшується.

2) Збільшимо обсяг даних. Для розрахунку візьмемо матрицю рангу 200 x 200. Кількість інформації: $8 \cdot 200 \cdot 200 = 320000$ байтів = 320 кбайтів. Результат наведений на Рис.10. Можна зробити висновок, що час обчислень значно зріс. Проте, зі збільшенням кількості процесорів його можна скоротити.

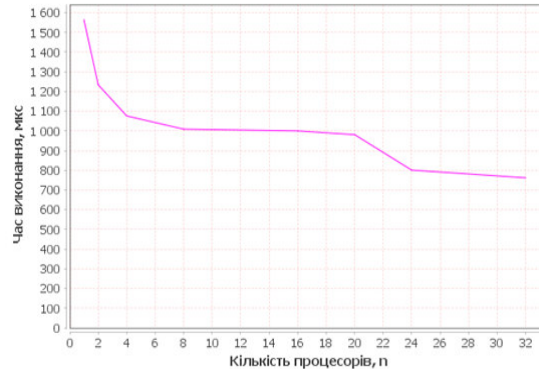


Рис. 10. Залежність часу виконання обчислень в залежності кількості обчислювальних процесів при розмірі матриці 200 x 200

3) Для розрахунку візьмемо матрицю рангу 1500 x 1500. Кількість інформації: $8 \cdot 1500 \cdot 1500 = 18000000$ байтів = 18 Мбайтів (Рис. 11).

З графіка видно, що час обчислень зменшується при збільшенні кількості процесорів.

Загальні результати щодо проведених експериментів наведені в таблиці 1.

З результатів проведених експериментів можна зробити висновок, що при збільшенні обсягів даних час обчислень постійно зростає. Для того, щоб підвищити продуктивність роботи системи, слід проводити розподілену обробку інформації. Наприклад, проведення обчислень над двохвимірним масивом даних, що представлений у вигляді матриці рангу 20, на звичайному нерозподіленому процесорі триває більше 20 мс. При використанні розподіленої системи, що складається з 32 процесорів, час обробки скорочується вдвічі.

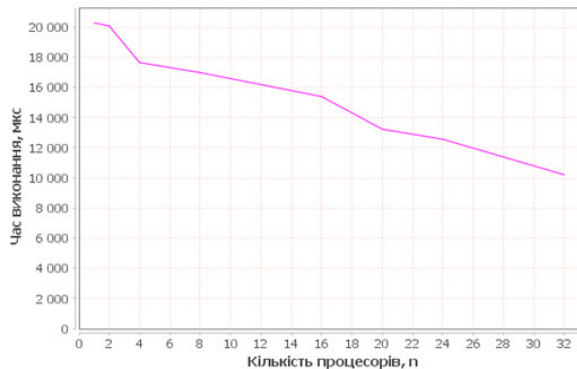


Рис. 11. Залежність часу виконання обчислень в залежності кількості обчислювальних процесів при розмірі матриці 1500 x 1500

Таблиця 1

Залежність часу виконання обчислень в залежності кількості обчислювальних процесів та розмірів матриці

К-ть процесорів	Розмір матриці $[n \times n]$, n		
	1500	200	20
	Час виконання, мкс		
1	20286	1564	18,1
2	20076	1234	16
4	17654	1076	15,7
8	16976	1009	15
16	15400	1000	12,5
20	13234	981	10,2
24	12567	801	10
32	10213	762	8,7

Подібні показники можна спостерігати і при інших обсягах вхідних даних. Так, для розміру матриці 200×200 час обробки скорочується від 1564 мкс на одному процесорі до 762 мкс на 32-х. Для найбільшого масиву даних рангу 1500, також можна добитися зменшення тривалості обробки вдвічі, застосовуючи розподілену систему з багатьма обчислювальними пристроями.

При обробці великих масивів даних необхідно застосовувати розподілені системи обчислень, причому зі значною кількістю процесорів.

Порівняльний аналіз систем обробки даних з використанням протоколу MPI та програмної моделі MapReduce

Обробка даних за допомогою технології Hadoop. Проведемо обчислення алгоритму SVD, обчислюючи різні обсяги даних в системі з використанням моделі MapReduce за допомогою технології Hadoop. Результати наведені в табл. 2 та на Рис. 12.

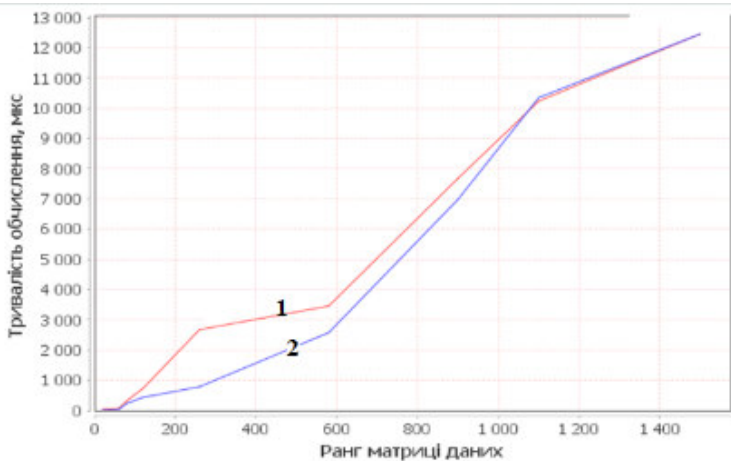


Рис. 12. Залежність часу виконання обчислень в залежності від розміру масиву обчислюваних даних в кластері Hadoop та порівняння технології Hadoop (крива 1) та протоколу MPI (крива 2)

Таблиця 2

Залежність часу виконання обчислень в залежності від розміру масиву обчислюваних даних в кластері

Розмірність матриці даних	Час виконання обчислень, мкс
20	48
60	67
80	323
120	726
260	2675
580	3456
900	7678
1100	10234
1500	12456

З отриманих результатів можна зробити висновок, що зі збільшення рангу матриці час обробки зростає. Очевидно, що системи з MPI ефективніше проводять обчислення, ніж з Hadoop. Але, при збільшенні обсягів даних різниця в часі не такою помітною. Враховуючи, що модель MapReduce (реалізована ніж допомогою Hadoop) значно простіша в реалізації за протокол обміну повідомленнями MPI, тому доцільно використовувати її в системах з досить великим обсягом даних.

Висновки

Отже, підсумовуючи вище викладене, варто відмітити результати досліджень:

1. Звичайні нерозподілені системи обробки даних неефективні для великого обсягів інформації через невисоку продуктивність обчислень. Пропонується використовувати розподілені системи, в яких використовують метод сингулярної декомпозиції даних, що дозволить зменшити обсяг оброблюваної інформації.
2. В результаті дослідження систем з використанням протоколу MPI та моделі MapReduce отримано залежності тривалості обчислень від кількості процесів, які свідчать про доцільність використання розподілених обчислень при обробці великих масивів даних. Також визначено, що розподілені системи із застосуванням моделі обчислень MapReduce працюють значно ефективніше ніж за протоколом MPI, особливо при великому обсягу даних.

Література

1. J. Dean, S. Ghemawat, «Mapreduce: Simplified data processing on large clusters», *ACM Commun.*, vol. 3, no. 5, May 2017, P. 5–6.
2. M. Rajathi, M. Ramaswami, «A Survey on Factorization Methods in MapReduce Environment», *International Journal of Computational Intelligence and Informatics*, vol. 7, no. 4, March 2018, P. 9–10.
3. H. Avron, S. Toledo, «LAPACK's least — squares solver», *SIAM*, 2010
4. S. Kim, C. Chu, «MapReduce for mashine learning on multicore», *International Journal of Computational Intelligence and Informatics*, vol. 6, no. 7, Juni 2018, P. 5–6.
5. Антонов А.С. Технологии параллельного программирования MPI и OpenMP: Учеб. пособие. — М., 2012. — С. 7–9.

6. Т.В. Борис, М.О. Алексєєв, «Порівняльний аналіз технології паралельного обчислення великих масивів даних MapReduce», *Second International Conference «Cluster Computing»*, Львів, 2013, с. 1–3.

7. J. Talbor, C. Kozyrakis, «Phoenix++: modular MapReduce for shared — memory systems», *International Journal of Computational Intelligence and Informatics*, vol. 2, no. 8, Februar 2016, P. 3–4.8, 2014, P. 10–14.

References

1. J. Dean, S. Ghemawat, «Mapreduce: Simplified data processing on large clusters», *ACM Commun.*, vol. 3, no. 5, May 2017, P. 5–6.

2. M. Rajathi, M. Ramaswami, «A Survey on Factorization Methods in MapReduce Environment», *International Journal of Computational Intelligence and Informatics*, vol. 7, no. 4, March 2018, P. 9–10.

3. H. Avron, S. Toledo, «LAPACK's least — squares solver», SIAM, 2010.

4. S. Kim, C. Chu, «MapReduce for mashine learning on multicore», *International Journal of Computational Intelligence and Informatics*, vol. 6, no. 7, Juni 2018, P. 5–6.

5. Antonov A.S. *Technologii paralelnogo programirivania MPI i OpenMP*: — M., 2012. — s. 7–9.

6. T.V. Boris, M.O.Alekseev, *Porivnialnyj analiz tehnologij paralelnogo obchyslennia velykych masyviv danyh MapReduce*, *Second International Conference «Cluster Computing*, Lviv, 2013, s. 1–3.

7. J. Talbor, C. Kozyrakis, «Phoenix++: modular MapReduce for shared — memory systems», *International Journal of Computational Intelligence and Informatics*, vol. 2, no. 8, Februar 2016, P. 3–4.8, 2014, P. 10–14.